대 한 민 국 특 허 청
KOREAN INTELLECTUAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto is a true copy from the records of the Korean Intellectual Property Office.

출 원 번 호 : 10-2002-0077596
Application Number PATENT-2002-0077596

출 원 년 월 일 : 2002년 12월 07일
Date of Application DEC 07, 2002

출 원 인 : 한국전자통신연구원
Applicant(s) Electronics and Telecommunications Research Institu

2003 년 01 월 08 일

특 허 청
COMMISSIONER

【서지사항】

【서류명】	특허출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	0001
【제출일자】	2002.12.07
【발명의 명칭】	마이크로 컨트롤러 소프트 아이피 내장용 롬 소프트 아이피의 생성 방법 및 이 방법을 실행시키기 위한 프로그램을 기록한 기록매체
【발명의 영문명칭】	METHOD FOR CREATING ROM SOFT IP USING IN THE MICRO CONTROLLER SOFT IP AND STORAGE MEDIA FOR PROGRAM OF PERFORMING THE SAME
【출원인】	
【명칭】	한국전자통신연구원
【출원인코드】	3-1998-007763-8
【대리인】	
【성명】	신영무
【대리인코드】	9-1998-000265-6
【포괄위임등록번호】	2001-032061-5
【발명자】	
【성명의 국문표기】	임태영
【성명의 영문표기】	LIM,Tae Young
【주민등록번호】	460301-1079413
【우편번호】	305-330
【주소】	대전광역시 유성구 지족동 열매마을아파트 505-201
【국적】	KR
【발명자】	
【성명의 국문표기】	곽명신
【성명의 영문표기】	KWAK,Myung Shin
【주민등록번호】	520102-1255429
【우편번호】	305-755
【주소】	대전광역시 유성구 어은동 한빛아파트 108-1602
【국적】	KR

【발명자】
【성명의 국문표기】 김종대
【성명의 영문표기】 KIM,Jong Dae
【주민등록번호】 540809-1110127
【우편번호】 302-724
【주소】 대전광역시 서구 관저동 대자연마을아파트 108-2105
【국적】 KR
【발명자】
【성명의 국문표기】 조양기
【성명의 영문표기】 CHO,Yang Ki
【주민등록번호】 700813-1052026
【우편번호】 361-772
【주소】 충청북도 청주시 흥덕구 분평동 주공7차아파트 707동 1301호
【국적】 KR
【발명자】
【성명의 국문표기】 송승완
【성명의 영문표기】 SONG,Seung Wan
【주민등록번호】 740504-1009717
【우편번호】 431-060
【주소】 경기도 안양시 동안구 관양동 1588-13 공작아파트 313-604
【국적】 KR
【발명자】
【성명의 국문표기】 김희석
【성명의 영문표기】 KIM,Hi Seok
【주민등록번호】 541223-1024417
【우편번호】 135-968
【주소】 서울특별시 강남구 대치1동 삼성아파트 102동 501호
【국적】 KR
【심사청구】 청구
【취지】 특허법 제42조의 규정에 의한 출원, 특허법 제60조의 규정에 의한 출원심사 를 청구합니다. 대리인
신영무 (인)

【수수료】

【기본출원료】	20	면	29,000 원
【가산출원료】	7	면	7,000 원
【우선권주장료】	0	건	0 원
【심사청구료】	9	항	397,000 원
【합계】	433,000 원		
【감면사유】	정부출연연구기관		
【감면후 수수료】	216,500 원		

【기술이전】

【기술양도】	희망
【실시권 허여】	희망
【기술지도】	희망

【첨부서류】 1. 요약서·명세서(도면)_1통

【요약서】

【요약】

본 발명은 마이크로 컨트롤러 소프트 아이피에 내장 할 수 있는 롬 소프트 아이피(ROM Soft IP)의 생성 방법 및 이 방법을 실행시키는 프로그램을 기록한 기록체에 관한 것으로서, 특히 MPU 코어 IP에 내장 할 수 있도록 롬 소프트 IP를 생성함으로서 IP 설계자가 용이하게 롬 내장 MPU 소프트 IP를 설계 할 수 있을 뿐만 아니라 MPU 코어 IP의 사용자도 프로그래밍 된 크기와 동일한 롬 프로그램 데이터들을 넣을 수 있는 롬 소프트 IP를 생성할 수 있다.

【대표도】

도 1

【색인어】

MPU, 소프트 아이피, Hex file, VHDL, 마스크롬

【명세서】

【발명의 명칭】

마이크로 컨트롤러 소프트 아이피 내장용 롬 소프트 아이피의 생성 방법 및 이 방법을 실행시키기 위한 프로그램을 기록한 기록매체 {METHOD FOR CREATING ROM SOFT IP USING IN THE MICRO CONTROLLER SOFT IP AND STORAGE MEDIA FOR PROGRAM OF PERFORMING THE SAME}

【도면의 간단한 설명】

도 1은 본 발명의 바람직한 실시예에 의한 롬 소프트 IP를 생성 방법과 MPU 코어 IP와 합쳐서 롬 내장 MPU 소프트 IP에 적용하는 방법의 흐름도를 도시한 도면이다.

도 2는 MPU 소프트 IP의 개략적인 구조 및 IP간의 연결관계를 설명하기 위한 개념도이다.

도 3은 도 1의 롬소프트 IP를 생성 방법의 헤더 파일(head.vhd)의 일예를 도시한 도면이다.

도 4는 도 1의 롬소프트 IP를 생성 방법의 테일 파일(tail.vhd)의 일예를 도시한 도면이다.

도 5는 도 1의 롬소프트 IP를 생성 방법의 핵사 파일(watch.hex)의 일예를 도시한 도면이다.

도 6a 및 도 6b는 상기 도 1의 롬 소프트 IP 생성 알고리즘의 세부 구현 단계들의 흐름을 도시한 도면이다.

도 7은 상기 도 1에서 제시된 롬 소프트 IP 생성 알고리즘을 수행하여 생성한 롬 소프트 IP 파일의 일예이다.

【발명의 상세한 설명】

【발명의 목적】

【발명이 속하는 기술분야 및 그 분야의 종래기술】

<8> 본 발명은 마이크로 컨트롤러 소프트 아이피에 내장할 수 있는 롬 소프트 아이피(ROM Soft IP)를 생성하는 방법에 관한 것으로서, 보다 상세하게는, MPU 코어 IP(MPU core IP)에 내장 할 수 있도록 롬 소프트 IP를 생성함으로써, IP 설계자가 용이하게 롬 내장 MPU(Micro Processor Unit) 소프트 IP를 설계 할 수 있을 뿐 아니라 MPU 코어 IP 사용자도 프로그래밍된 크기와 동일한 롬 프로그램 데이터(ROM program data)들을 넣을 수 있는 롬 소프트 IP를 생성하는 방법에 관한 것이다.

<9> 이하, 본 명세서를 통해서 언급되는 용어들에 대해 정의한다.

<10> "마이크로 컨트롤러"는 Micro Processor Unit를 의미하며, 이하 MPU로 표기하고, "아이피(IP)"는 Intellectual Property의 약칭으로서, 반도체 설계자산을 의미하며, 이하 IP로 표기하기로 한다. 또한 "MPU 코어 IP"는 롬을 내장하지 않은 MPU IP를 의미하며, "MPU 소프트 IP"는 롬을 내장한 MPU IP를 의미하고, 반도체 제조 공정의 디자인 룰(Design Rule)에 무관하게 칩으로 구현할 수 있도록 전자회로 설계 언어로 기술하는 가상부품(Virtual Component)으로서, 마이크로 프로세서 기능의 반도체 설계자산(Semiconductor IP)을 의미한다.

<11> 한편, "소프트 IP"는 전자회로를 구현할 수 있도록 VHDL(Very High speed Description Language), Verilog 등의 전자회로 설계언어로 표현된 반도체 설계자산을

의미하고, "IP 설계자(IP Designer)"는 반도체 설계자산을 창출하는 사람을 의미하며, "IP 사용자(IP User)"는 상기 IP들을 활용하여 어떤 시스템을 창출하는 사람을 의미하기로 한다. 또한, "프로그램 데이터(Program Data)"는 롬 내부에 프로그래밍 되어있는 MPU 구동용(Operating) 핵사코드(Hex code)들을 의미한다. "하드 IP(Hard IP)"는 전자회로의 연결정보들로 구성되어 특정 칩 제조회사에서만 칩 제조가 가능한 IP를 의미한다.

<12> 이하, 종래 기술에 따라서 MPU IP를 설계하는 방법 및 이에 따른 단점들을 설명한다.

<13> 일반적으로 임의의 MPU 칩을 구동시키기 위해서는 핵사 코드 형태로 되어 있는, 해당 MPU용 명령어 집합(Instruction Set)인 오퍼레이션 코드(OPeration code, OP code)들을 활용하여 해당 MPU의 기능을 발휘할 수 있도록 프로그래밍 해서, 해당 MPU의 내부 또는 외부의 읽기 전용 기억소자(Memory Device)(이하 "기억소자" 라고 약칭하기로 함. 또한 "마스크롬(Mask ROM)" 과 "프로그램 메모리 블럭 및 기억소자 블럭" 도 동일한 블럭을 지칭하는 것으로 함)에 기록해 넣어야 한다. 따라서 롬 내장 MPU 칩은 제조가 완료된 후에도 사용자들이 외부에서 프로그래밍 할 수 있도록 이피롬(EPROM : Erasable Programmable Read Only Memory)이나 이투피롬(EEPROM : Electrically Erasable Programmable Read Only Memory) 등의 불휘발성(Non-volatile) 프로그래밍 가능 읽기전용 기억소자(Programmable Read Only Memory Device)를 MPU 칩 내부에 내장하여 설계를 해야한다.

<14> 종래 기술에 의한 MPU IP 구현방법을 더욱 상세히 설명하기 위해서, 표 1에는 종래 기술1 내지 5의 MPU IP 구현방법의 경우, 각각의 MCU부분, 기억소자 부분, IP 형태 및 설계체계 및 주체에 대해서 대략적으로 기술하였다.

<15> 【표 1】

구분	MPU 구현				기억소자 내장 MPU IP 구현	
	MCU 부분		기억소자 부분		IP 형태	설계 체계 및 주체
	설계 방법	설계 주체	구현 공정 기술	구동용 프로그램		
종래기술 1	회로도 또는 설계언어	IP 설계자	이피롬(EPROM) 또는 이투피롬(EEPROM)	사용자가 상품화된 칩의 외부에서 롬 프로그램 써넣음	하드 IP (회로도 위주)	1. MCU -> IP설계자 2. 기억소자 -> 칩 제작자 (반도체 회사)
종래기술 2					소프트 IP (설계언어 위주)	1. MCU -> IP설계자 2. 기억소자 -> IP 설계자
종래기술 3	회로도 또는 설계언어	IP 설계자	마스크롬	사용자가 칩 완성 전에 롬 프로그램 제공	하드 IP (회로도 또는 설계언어)	1. 롬 프로그램 -> 롬 코드 -> IP 사용자 2. 롬 코드 -> 롬 패턴 -> IP 설계자 3. 롬 패턴 -> CAD 툴 -> 마스크 롬 -> IP 설계자 또는 칩 제작자
종래기술 4					소프트 IP (설계언어 위주)	상기와 유사
종래기술 5						1. 롬 프로그램 -> 롬 코드 -> IP 사용자 2. 롬 코드 -> VHDL 파일로 변환시키고, 기 작성한 MPU 코어 IP용 소스파일도 수정 -> IP 설계자

<16> 하지만, 표 1에서 상술한 바와 같은 종래 기술들은 그 각각이 다음과 같은 문제점들을 가지고 있다.

<17> 먼저, 종래기술 1에 의한 이투피롬 내장 MPU를 하드 IP(Hard IP)로 설계하는 방법은, MPU 코어부분은 IP 설계자가 설계하고, 이투피롬 부분은 칩 제작자가 완성하게 된다. 따라서 칩 제작공정 회사가 달라지면 공정조건을 수정해서 다시 설계 해야 하는 단점이 있다

<18> 종래 기술 2의 이투피롬 내장 MPU를 소프트 IP(Soft IP)로 설계하는 방법은, 칩 제조공정이 매우 복잡하고 공정기술에 변수가 많기 때문에 칩으로 제조하기가 거의 불가능하다. 따라서 이투피롬 내장 MPU를 소프트 IP로 설계할 경우에는 롬을 내장하지 않고 MPU 코어부분만 설계 해야하는 단점이 있다.

<19> 종래 기술 3과 같이, MPU 칩의 사용자로부터 사전에 프로그래밍 데이터를 수령해서 롬 패턴(ROM pattern)으로 바꾼 후 이 패턴에 해당하는 마스크를 이용하여 제작하는 마스크롬 내장 MPU를 하드 IP(Hard IP)로 설계하는 방법은, 코어부분은 설계자가 설계하고, 마스크롬 부분의 설계는 IP 사용자로부터 사전에 프로그래밍 데이터를 수령하고, 칩 제작자로부터 공정정보를 받아서 CAD(Computer Added Design) 툴(Tool)에 의존하여 완성하게 된다. 따라서 칩 제작공정 회사가 달라지거나 툴이 달라지면 공정조건 및 설계방법을 달리해서 다시 설계 해야하는 단점이 있고, MPU 하드 IP의 사용자가 IP 상태에서는 프로그래밍 데이터를 수정할 수 없다는 단점이 있다.

<20> 종래 기술 4에 의한 마스크롬 내장 MPU를 소프트 IP(Soft IP)로 설계하는 방법은, IP 사용자로부터 사전에 프로그래밍 데이터를 수령하고, IP 설계자가 특정 CAD 툴(Tool)을 이용하여 마스크롬 코드를 패턴으로 변환시켜서 4K 바이트 등으로 특정된 메모리 구

역에 데이터를 각인하는 방법으로 롬 블럭의 전자회로를 설계하고 MPU 코어 부분의 전자회로를 설계한 후 또 다른 CAD 툴을 이용하여 결합시키는 방식으로 MPU 소프트 IP를 설계한다. 따라서 CAD 툴이 달라지면 설계방법 달리해서 다시 설계 해야하는 단점이 있고, 잉여의 기억소자가 생성되며, MPU 소프트 IP의 사용자가 IP 상태에서는 프로그래밍 데이터(마스크롬의 데이터를 의미 함)를 수정할 수 없다는 단점이 있다.

<21> 종래 기술 5의 마스크롬 내장 MPU를 소프트 IP(Soft IP)로 설계하는 방법은, IP 설계자가 전자회로 설계 언어로서 MPU 코아만 설계하고, IP 사용자가 상용의 MPU 프로그래밍 툴에서 프로그래밍 해놓은 구동용 핵사코드(마스크롬의 데이터를 의미 함)들을 IP 설계자가 수작업을 통해 메모리용 VHDL 파일로 변환시키고, 기 작성한 MPU 코어 IP용 소스파일에서 프로그램 메모리의 크기를 결정하고, 어드레스 버스에 관련된 소스파일 부분을 수정하여야 한다. 따라서, IP 설계자는 CAD 툴을 이용하여 수정된 IP의 신뢰성을 검증하기 위해서 각각 추출된 블럭들을 결합하여 합성 및 시뮬레이션 등의 검증 작업을 다시 거쳐야 하는 번거로움이 있고, 많은 수작업으로 인하여 설계 시간이 매우 길어지며 그 방법이 매우 복잡하다는 단점이 있다. 또한, MPU 소프트 IP의 사용자가 IP 상태에서, 프로그래밍 데이터를 수정할 수 없다는 단점이 있다

<22> 따라서, 상술한 바와 같은 종래기술에 의한 방법들을 종합해보면, 이투피롬을 내장하는 하드 IP로는 칩 제작공정 회사가 달라지면 공정조건을 수정해서 다시 설계 해야하는 문제점이 있고, 마스크롬을 내장하는 하드 IP로는 칩 제작공정 회사가 달라지거나 CAD 툴이 달라지면 공정조건 및 설계방법 달리해서 다시 설계 해야하는 문제점이 있고,

MPU 하드 IP의 사용자가 IP 상태에서는 프로그래밍 데이터를 수정할 수 없다는 문제점이 있다.

<23> 또한, 이투피롬을 내장하는 소프트 IP는 칩 제조공정이 매우 복잡하고 공정기술에 변수가 많기 때문에 코어부분만 설계하는 문제점이 있고, 마스크롬을 패턴 형식으로 내장하는 소프트 IP는 CAD 툴이 달라지면 설계방법을 달리해서 다시 설계 해야하는 문제점이 있고, 잉여의 기억소자가 생성되며, MPU 소프트 IP의 사용자가 IP 상태에서는 프로그래밍 데이터를 수정할 수 없다는 문제점이 있다. 또한, 마스크롬을 전자회로 설계 언어로 내장하는 소프트 IP는 많은 수작업으로 인하여 설계 시간이 매우 길어지고 그 방법이 매우 복잡하며, MPU 소프트 IP의 사용자가 IP 상태에서는 프로그래밍 데이터를 수정할 수 없다는 문제점이 있다.

【발명이 이루고자 하는 기술적 과제】

<24> 따라서, 본 발명은 상기 문제점을 해결하기 위하여 안출된 것으로써, MPU에 내장하는 마스크롬의 데이터들을 소프트 IP 형태가 되도록 생성하기 위하여, MPU 구동용 프로그램 롬 코드(마스크롬의 데이터를 의미 함)를 VHDL 등의 전자회로 설계언어 파일로 자동변환 시키되, 롬 콤포넌트(ROM Component) 파일 형태로 생성함으로서 MPU 소프트 IP의 회로 합성과정을 간략화 하는데 유용하게 적용될 수 있고, 프로그래밍 되어있는 데이터 크기 만큼만을 변환시킴으로서, 특정한 크기를 갖고있는 마스크롬보다 칩의 크기를 최소화하는데 유용하게 적용될 수 있는 롬 소프트 IP의 생성 방법을 제공하는 것이다.

<25> 본 발명의 다른 목적은, MPU 프로그래밍 데이터(마스크롬의 데이터를 의미 함)를 MPU 소프트 IP의 사용자도 용이하게 변환할 수 있고, MPU의 최상위 어드레스까지 수용함으로서 외장 롬 용량까지 최대한 코딩할 수 있는, 독립 파일 형태의 롬 소프트 IP를 생성하는 방법을 제공하는 것이다.

<26> 또한, 본 발명의 또다른 목적은 MPU 소프트 IP의 설계자는 MPU 코어 IP에 롬을 내장한 MPU 소프트 IP를 용이하게 또한 최대 롬 용량까지 수용하도록 설계할 수 있으며, MPU 소프트 IP의 사용자는 MPU 프로그래밍 데이터를 독자적으로 변경 및 수정 또는 MPU 코어 IP 만 활용할 수 있도록 삭제할 수 있는 것이다.

<27> 본 발명의 또다른 목적은, 반도체 칩 제조회사의 롬 제조 공정기술 보유여부에 무관하도록 전자회로 설계 언어로 묘사함으로서 어떤 반도체 칩 제조회사에서든 칩으로 구현할 수 있는 롬 소프트 IP의 생성 방법을 제공하는 것이다.

【발명의 구성 및 작용】

<28> 상기 목적을 달성하기 위한 본 발명의 일측면은 (a) 롬 소프트 IP의 초기 정보를 기술하고 롬 어드레스와 명령어들을 묘사할 구문을 지정하는 헤더 파일, 종료 정보들을 기술한 테일 파일 및 롬 소프트 IP의 행위를 묘사할 빈 파일을 작성하고, MPU 프로그램 메모리(Program Memory)용 핵사 파일을 선정하는 단계와, (b) 빈 파일 내부에, 상기 헤더 파일을 복사하고 상기 프로그램 메모리용 핵사 파일에서 ASCII 문자로 구성된 어드레스와 명령어를 통해서 시작번지와 명령어를 전자회로 설계언어로 변환하고, 테일 파일을 복사하여 롬 코드 변환 프로그램을 안출하는 단계와, (c) 롬 코드 변환 프로그램을 실행시켜서 롬 소프트 IP용 파일을 생성하는 단계를 포함하여 이루어지는 롬 소프트 IP의 생성 방법을 제공한다.

<29> 한편, 프로그램 메모리용 핵사 파일에서 ASCII 문자로 구성된 어드레스와 명령어를 통해서 시작번지와 명령어를 전자회로 설계언어로 변환하는 과정은, 어드레스와 명령어를 분석하여 각각 2진수로 변환한 후 시작번지와 명령어를 전자회로 설계언어로 써넣을 수 있고, 롬 소프트 IP는 VHDL, Verilog를 포함하는 전자회로 설계언어들 중 어느 하나로 묘사될 수 있으며, 롬 소프트 IP은 마스크롬을 대체하여 MPU에 내장할 수 있다.

<30> 또한, 헤더파일에는 IP에 적용할 라이브러리, IP의 명칭 및 입출력 신호를 포함하는 초기정보, 롬 어드레스와 이 어드레스에 있는 명령어들을 묘사할 구문을 지정하되 전자회로 설계언어로 작성되고, 테일파일에는 롬의 마지막 데이터를 쓰고 종료문을 지정하되, 전자회로 설계언어로서 작성될 수 있다.

<31> 또한, 상술한 명령어를 전자회로 설계언어로 변환하는 과정은 ASCII 문자로 구성된 핵사 파일의 명령어 개수를 산출하되, 문자를 정수로 변환하는 함수를 이용하여 10진수를 산출하는 것이 바람직하다.

<32> 전자회로 설계 언어로 기술한 상기 롬 소프트 IP는 MPU 코어 IP와 병합해서 부품처럼 내장하는 것도 가능하며, 상기 롬 소프트 IP 생성방법은 생성된 롬소프트 IP와 상기 MPU 코어 IP를 병합하여, CAD툴로 회로합성 및 검증을 수행하는 단계를 추가로 포함가능하다.

<33> 본 발명의 다른 측면은 상술한 바와 같은 롬소프트 IP의 생성방법을 실행시키기 위한 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체를 제공한다.

<34> 이하, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자가 본 발명의 기술적 사상을 용이하게 실시할 수 있을 정도로 상세히 설명하기 위하여, 본 발명의 가장 바람직한 실시예를 첨부된 도면을 참조하여 설명하기로 한다.

<35> 본 발명은 롬 소프트 IP 생성 알고리즘이라고 명명되는 롬의 데이터들을 전자회로 설계언어 파일로 변환시켜서 새로운 롬 소프트 IP를 생성할 수 있는 알고리즘을 제안한다. "롬 소프트 IP 생성 알고리즘"이란, MPU 소프트 IP에 내장하거나 외부에 부착해야 하는 특정한 용도와 크기를 갖고 있는 마스크롬의 데이터를, Visual C 등의 프로그래밍 소프트웨어로 제작할 수 있는 롬 코드 변환 프로그램을 안출하고 이를 실행시킴으로서, 전자회로 설계 언어 파일로 변환시켜 MPU 소프트 IP와 병합하거나 삽입할 수 있도록 독립된 롬 소프트 IP를 생성하는 알고리즘이다.

<36> 특히 MPU의 용도에 따라서 다르게 프로그래밍 된 마스크롬 코드가 담겨진 핵사 파일을 읽어들여서, VHDL 등의 전자회로 설계언어 파일로 변환시키는 방법에 의하여 마스크롬 데이터 크기와 동일한 롬 콤포넌트 파일을 생성함으로서, 칩의 크기를 최소화하는데 유리하고 MPU의 최상위 어드레스도 수용이 가능하여 외장 롬부분도 내장할 수 있는 다양한 크기의 롬 소프트 IP를 생성할 수 있고, MPU 코어 IP의 설계자 뿐만 아니라 사용자도 롬을 용이하게 내장할 수 있으며, 또한, 전자회로 설계 언어로 묘사함으로서 반도체 칩 제조회사의 롬 제조 공정기술 보유여부에 무관하게 칩으로 구현할 수 있으며, 마스크롬 형태의 메모리 블럭이 필요한 여러 종류의 SoC(System on a Chip)를 설계할 때에도, IP 사용자가 용이하게 활용할 수 있는 특징을 가진 롬 소프트 IP 생성 알고리즘이다.

<37> 이하, 첨부한 도면들을 참조하여 본 발명의 바람직한 실시예에 의한 롬 소프트 IP를 생성 방법과 MPU 코어 IP와 합쳐서 롬 내장 MPU 소프트 IP에 적용하는 방법에 대하여 도 1의 흐름도를 참조하여 구체적으로 설명한다.

<38> 먼저, MPU 소프트 IP의 개략적인 구조 및 IP 간의 연결체계를 도 2를 참조하여 설명하면, 도 2는 MPU 소프트 IP의 개략적인 구조 및 IP간의 연결관계를 설명하기 위한 개념도이다. 다만, 롬 내장 MPU 소프트 IP(10)에 적용하는 체계와 파일 명칭들은 본 발명에 대한 설명의 편의를 제공하기 위한 일예이다.

<39> 도 2를 참조하면, Inprom.vhd로 명명한 롬소프트 IP(20)에 16비트 어드레스 버스(addr)가 입력되면 8비트 명령어 버스(dataout) 출력 신호를 출력하도록 구성되어 있고, 0~64KB 크기의 MPU 구동용 프로그램을 코딩할 수 있는 롬 소프트 IP(20)가 생성된다. 롬소프트 IP(20)는 ET8031.vhd로 명명한 MPU 코어 IP(30)와 병합되어 CAD툴로 회로합성 및 검증 과정을 거친 후 롬 내장 MPU 소프트 IP(10)합쳐서 MPU 소프트 IP, CAD 툴로 회로를 합성 및 검증하는 과정을 거쳐서, ET8051.vhd로 명명한 롬 내장 MPU 소프트 IP(10)에 적용된다.

<40> 다음으로, 도 1의 흐름도를 상세히 설명한다.

<41> (S1 단계)

<42> 롬 소프트 IP(20)는 워크스테이션이나 PC(Personal Computer) 환경에서 전자회로 설계언어로서, 롬 소프트 IP의 초기 정보들을 기술한 헤더 파일과 종료 정보들을 기술한 테일 파일을 작성하여 구비하고, 새로운 롬 소프트 IP를 묘사하기 시작할 빈 파일을 구

비하며, 롬 소프트 IP로 변환시킬 대상인, ASCII 문자로 구성된 MPU의 프로그램 메모리용 헥사 파일을 준비한다(S1).

<43> 롬 소프트 IP(20)를 좀 더 구체적으로 설명하면, 도 3은 헤더 파일(head.vhd)의 구조를 도시한 도면이다. 헤더 파일(head.vhd)의 내부에는 IP에 적용할 라이브러리(A)와 IP의 명칭과 입출력 신호(B) 등에 관한 롬 소프트 IP(20)의 초기 정보들을 지정하고, 롬 어드레스와 이 어드레스에 담겨있는 명령어들을 묘사할 구문(C)을 지정하되, 전자회로 설계언어로서 작성하여 구비한다. 한편, 헤더 파일(head.vhd)은 VHDL파일 작성 형식의 국제표준 중에서 헤더파일부분만 떼어낸 것이다.

<44> 또한, 테일 파일은 tail.vhd 라고 명명하여 도 4에 예시하였다. 도 4에서 테일 파일 내부에는 롬의 데이터를 쓰고(D), 종료문(E)을 지정하되, 전자회로 설계언어로서 작성하여 구비한다. 또한, 새로운 롬 소프트 IP는 inprom.vhd 라고 명명하여, 전자회로 설계언어로서 작성할 빈 파일을 준비해 놓는다. 한편, 테일 파일(tail.vhd)은 VHDL파일 작성 형식의 IEEE 국제표준 중에서 헤더파일부분만 떼어낸 것이다.

<45> 또한, ASCII 문자로 구성된 MPU의 프로그램 메모리용 헥사 파일을 준비한다. 본 예에서는 watch.hex 라는 명칭의 시계기능을 구동시키는 파일을 준비하여 도 5에 예시하였다. 도 5에서의 핵사 파일(watch.hex)은 MPU의 사용자가 프로그래밍 하게 되며, 파일 내부에는 특정된 규칙과 의미가 있다. 이를 좀 더 상세히 설명한다. 헥사 파일(watch.hex)의 첫번째행의 제1 문자(:, a)는 행의 시작을 의미한다. 또한 제2~3 문자(03, b)는 이 행에 포함된 OP 코드의 수를 16진수로 표현한 것으로서 본 예의 경우는 OP 코드가 3개 있다는 의미이다. 제4~7 문자(0000, c)는 이 행의 시작 번지를 표현한 것으로서 본 예에서는 16진수로 0000 번지임을 나타낸다. 제8~9 문자(00, d)는 마지막 라인

표시로서 마지막일 경우에만 01로 표시한다. 제10~11 문자(02, e), 제12~13 문자(1E, f), 제14~15 문자(DD, h)들은 이 행에 포함된 3개의 OP 코드를 의미한다. 또한 제14~15 문자(DD, h)는 체크썸으로서 행의 마지막에 표시되어있다. 핵사 파일들은 사용자가 어떤 용도로 MPU를 사용할 것인가에 따라서 코딩이 달라지며, 길이도 달라진다.

<46> (S2 단계)

<47> 다음으로, 롬 소프트 IP(20)는 PC 환경에서 Visual C 등의 프로그래밍 소프트웨어로서 RomConv.exe라고 명명한 롬 코드 변환 프로그램을 안출하는데, 상기에서 준비한 새로운 빈칸의 롬 소프트 IP 파일 내부에 상술한 헤더 파일(head.vhd)을 복사해 넣고, 프로그램 메모리용 핵사 파일(watch.hex)을 읽어 들여서, 프로그래밍 소프트웨어의 함수들을 이용하여, 롬 어드레스와 데이터를 VHDL 코드로 변환시켜서 헤더 파일이 복사된 롬 소프트 IP 파일에 기입하도록 한다(S2).

<48> 이와 같은 롬 코드 변환 프로그램은 일반적인 설계 지식을 갖고있는 사람이 프로그래밍 소프트웨어를 이용하여 스크립트 파일 또는 뱃치 파일 등을 적절히 혼합하여 수행시키도록 작성하는 것에 의하여 구현이 가능하다. 이하, 도 6을 참조하여 세부 알고리즘의 구현 방법을 흐름도의 일예를 설명한다.

<49> 롬 코드 변환 프로그램을 프로그래밍하기 위해서는, 먼저 MPU의 프로그램 메모리용 원시 파일을 컴파일하여 추출된, ASCII 문자로 구성된 롬 코드용 HEX 파일 (F1)(Watch.hex)의 내용을 전부 읽어서 임의의 변수(FBuf)에 저장하는 단계를 프로그램한다(S201). 이 후, 새로운 빈칸의 롬 소프트 IP 파일(F3)(Inprom.vhd)의 상부에 미리 작성한 헤더 파일(F2)(head.vhd)을 복사하여 써넣도록 프로그래밍하고(S202), 프로그램 메모리의 크기 만큼 어드레스가 증가되었는지를 확인하는 변수를 MaxAddr로 하여 "0"

으로 설정하도록 프로그래밍한다(S203). 다음으로, FBuf에 저장된 프로그램 메모리용 핵사 파일의 제1 행을 읽고 난 후 LBuf 라고 명명한 변수에 저장하도록 프로그래밍 한다(S204). 이 후, LBuf의 제2~3 문자(b)를 읽어서 변수 tempStr에 저장하고, 변수 tempStr과 문자 하나를 정수로 변환하는 함수 asc2Num을 이용하여 다음 식 1로 명령어 개수의 변수 N을 준비한다.

<50> N = asc2Num(tempStr(0)(제2 문자, 0이며 아스키 코드값은 48임)) * 16 + asc2Num(tempStr(1)(제3 문자, 3이며 아스키 코드값은 51임)) ---------- (1)

<51> 따라서, 식 1로 제2~3 문자(b)인 03을 아스키값인 3으로 변환하도록 프로그래밍 함으로서 N의 숫자를 준비하고(S205), 시작 번지를 분석해서 StAddr 변수에 저장하며(S206), 시작번지 "StAddr"가 프로그램 메모리 증가 변수 "MaxAddr" 보다 큰지 여부를 판단하여(S207), 큰 경우에는 "StAddr"와 "N"을 합산하여 "MaxAddr"를 증가시키고(S208), 크지 않은 경우에는 "StAddr"을 어드레스(Addr)로 간주하고 증가 변수 "Cnt"를 "0"으로 한다(S209).

<52> 다음으로, 1개의 명령어를 분석해서 OP 코드를 준비하여(S210), 이 OP 코드를 2진수로 변환하고(S211), 어드레스(Addr)와 2진수로 변환한 OP 코드를 VHDL 파일로 롬소프트 IP(inprom.d)의 헤더코더 및 롬데이터 코드(F4)로 써넣는다(S212). 그 후, 어드레스와 증가 변수 "Cnt"를 각1개씩 증가시키고(S213), 상기의 명령어 개수 N이 명령어 증가 변수 "Cnt"와 동일한지를 판단하여(S214) N이 마지막 명령어 증가 변수가 아니면 그 다음 명령어를 분석하도록 S210단계로 궤환시키며, 마지막 명령어 계수이면 파일 끝인지를 판단하여(S215), 끝이 아니면 다음 행을 읽도록 S204단계로 궤환하고, 끝이면 프로그램 메모리의 크기를 분석하고(S216), 테일 파일(tail.vhd)(F5)을 이용하여 VHDL 파일로 헤

더파일, 롬 데이터 파일 및 테일 파일을 갖는 inprom.vhd(F6)에 마무리 코드를 써넣고(S217), 종료하도록 프로그래밍 한다(S218).

<53> 한편, 상술한 롬 코드용 HEX 파일(watch.hex)은 반드시 MPU의 프로그램 메모리용 원시 파일을 컴파일하여 추출된 것일 필요는 없다. 명령어들을 데이터로만 표현한 롬 코드용 HEX 파일로도 변환이 가능하며, 이 경우에는 구체적 실시예에서 시작 어드레스를 자동으로 증가시키는 방법을 취하면 된다.

<54> (S3 단계)

<55> 다시 도 2로 돌아가서, S2단계 이후의 과정을 설명하면, 롬 소프트 IP(20)는 상기에서 안출한 RomConv.exe 라고 명명한 롬 코드 변환 프로그램을 워크스테이션이나 PC 환경에서 실행시킴으로서, 도 6a 및 도 6b에 도시한 바와 같이 Inprom.vhd 이라고 명명한 롬 소프트 IP 파일(도 6b의 F7)을 생성하게 된다. 한편, 상기와 같은 요령으로 롬 코드 변환 프로그램을 한번만 작성하면, 상기 롬 코드용 HEX 파일(watch.hex)만 교체하여 실행시킴으로서, 동일 종류의 MPU에 적용할 수 있는 새로운 롬 소프트 IP(Inprom.vhd)용 파일을 생성할 수 있고, 상기 헤더 파일과 테일 파일 및 롬 코드용 HEX 파일을 변화시키면 다른 종류의 MPU에도 적용하여 새로운 롬 소프트 IP용 파일을 생성할 수 있다. 상술한 바와 같은 방법으로 생성한 롬 소프트 IP 파일의 일예를 도 7에 도시하였다. 생성된 롬 소프트 IP 파일은 헤더코드가 복사된 부분(A, B 및 C), 생성된 롬데이터 코드(F4) 부분 및 테일코드(D 및 E)부분을 포함한다. 다만, 도 7에서는 이와 같은 코드들 중 헤더코드가 복사된 부분(A, B 및 C), 생성된 롬데이터 코드(F4) 부분을 도시하고 있다.

<56> (S4 단계 이후)

<57> 상술한 방식으로 생성된 롬 소프트 IP(Inprom.vhd)는 전자회로 설계 언어로 기술한 파일이므로, 도 1에 도시한 MPU 코어 IP(30)와 병합해서 부품처럼 내장 할 수가 있어서, 일반적인 CAD 툴로 회로 합성 및 검증절차를 거치면(S4), 롬 내장 MPU 소프트 IP 용이하게 활용할 수 있게 된다.

<58> 본 발명의 기술적 사상은 상기 바람직한 실시예에 따라 구체적으로 기술되었으나, 상기한 실시예는 그 설명을 위한 것이며 그 제한을 위한 것이 아님을 주의하여야 한다. 또한, 본 발명의 기술 분야의 통상의 전문가라면 본 발명의 기술 사상의 범위 내에서 다양한 실시예가 가능함을 이해할 수 있을 것이다.

<59> 본 발명이 롬 소프트 IP의 생성 방법을 위주로 설명되어 있지만, 본 기술 분야의 당업자들이라면 본 발명의 메커니즘이 다양한 유형의 명령어들을 포함하여 프로그래화할 수 있을 것이고, 이러한 프로그램을 기록한 컴퓨터 기록매체 형태로 배포될 수 있다. 컴퓨터 판독가능한 기록매체의 예로서는 하드디스크, 플로피 디스크, 하드 디스크 드라이브 및 시디롬 같은 기록형 매체가 포함될 수 있다.

【발명의 효과】

<60> 상술한 바와 같이, 생성하고자하는 롬 소프트 IP의 초기 정보들을 기술한 헤더 파일과 종료 정보들을 기술한 테일 파일을 작성하여 구비하고, 상기 롬 소프트 IP의 행위를 묘사할 빈 파일을 구비하며, 프로그램 메모리용 핵사 파일을 읽어들이고, 롬 코드 변환 프로그램을 실행시킴으로서, 마스크롬을 대체하여 MPU에 내장할 수 있는 롬 소프트

IP의 설계시 유용하게 적용될 수 있으며, 내장 롬을 포함하는 다양한 시스템칩(SoC, System on a Chip)을 설계할 때에도 널리 이용할 수 있다.

【특허청구범위】

【청구항 1】

롬 소프트 IP의 생성 방법에 있어서,

(a) 상기 롬 소프트 IP의 초기 정보를 기술하고 롬 어드레스와 명령어들을 묘사할 구문을 지정하는 헤더 파일, 종료 정보들을 기술한 테일 파일 및 상기 롬 소프트 IP의 행위를 묘사할 빈 파일을 작성하고, 상기 MPU 프로그램 메모리(Program Memory)용 핵사 파일을 선정하는 단계;

(b) 상기 빈 파일 내부에, 상기 헤더 파일을 복사하고 상기 프로그램 메모리용 핵사 파일에서 ASCII 문자로 구성된 어드레스와 명령어를 통해서 시작번지와 명령어를 전자회로 설계언어로 변환하고, 상기 테일 파일을 복사하여 롬 코드 변환 프로그램을 안출하는 단계; 및

(c) 상기 프로그램을 실행시켜서 롬 소프트 IP용 파일을 생성하는 단계를 포함하여 이루어지는 것를 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 2】

제 1 항에 있어서,

상기 프로그램 메모리용 핵사 파일에서 ASCII 문자로 구성된 어드레스와 명령어를 통해서 시작번지와 명령어를 전자회로 설계언어로 변환하는 상기 과정은,

상기 어드레스와 명령어를 분석하여 각각 2진수로 변환한 후 시작번지와 명령어를 전자회로 설계언어로 써넣는 것을 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 3】

제 1 항에 있어서,

상기 롬 소프트 IP는 VHDL, Verilog를 포함하는 전자회로 설계언어들 중 어느 하나로 묘사된 것을 특징으로 하는 롬 소프트 IP를 생성하는 방법.

【청구항 4】

제 1 항에 있어서,

상기 롬 소프트 IP은 마스크롬을 대체하여 MPU에 내장할 수 있는 것을 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 5】

제 1 항에 있어서,

상기 헤더파일에는 IP에 적용할 라이브러리, IP의 명칭 및 입출력 신호를 포함하는 초기정보, 롬 어드레스와 이 어드레스에 있는 명령어들을 묘사할 구문을 지정하되 전자회로 설계언어로 작성되고,

상기 테일파일에는 롬의 마지막 데이터를 쓰고 종료문을 지정하되, 전자회로 설계언어로서 작성되는 것을 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 6】

제 1 항에 있어서, 상기 명령어를 전자회로 설계언어로 변환하는 과정은 ASCII 문자로 구성된 핵사 파일의 명령어 개수를 산출하되, 문자를 정수로 변환하는 함수를 이용하여 10진수로 산출하는 것을 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 7】

제 1 항에 있어서, 전자회로 설계 언어로 기술한 상기 롬 소프트 IP는 MPU 코어 IP와 병합해서 부품처럼 내장하는 것을 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 8】

제 7 항에 있어서,

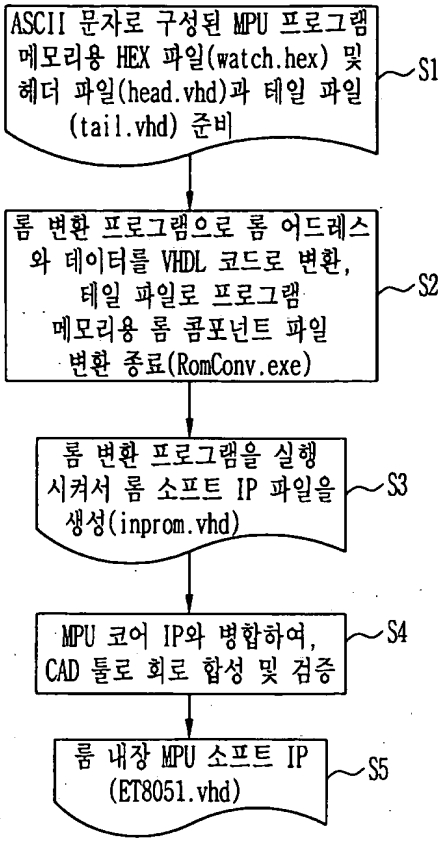
상기 생성된 롬소프트 IP와 상기 MPU 코어 IP를 병합하여, CAD툴로 회로합성 및 검증을 수행하는 단계를 추가로 포함하는 특징으로 하는 롬 소프트 IP의 생성 방법.

【청구항 9】

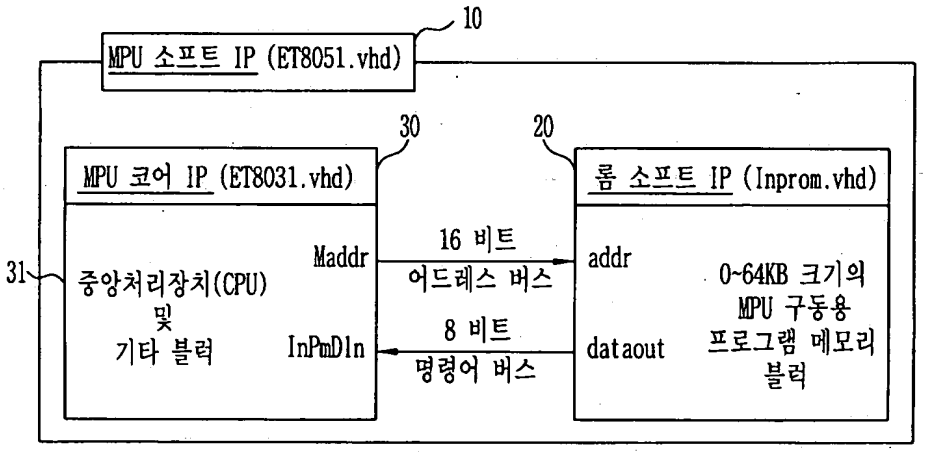
제 1 항 내지 제 8 항 중 어느 하나의 항에 의한 롬 소프트 IP의 생성방법을 실행시키기 위한 프로그램을 기록한 컴퓨터 판독 가능한 기록 매체.

【도면】

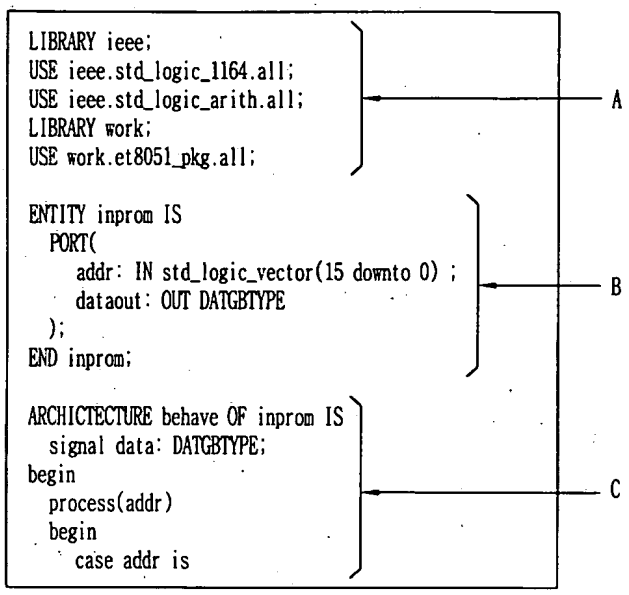
【도 1】


ASCII 문자로 구성된 MPU 프로그램 메모리용 HEX 파일(watch.hex) 및 헤더 파일(head.vhd)과 테일 파일 (tail.vhd) 준비
S1
롬 변환 프로그램으로 롬 어드레스와 데이터를 VHDL 코드로 변환, 테일 파일로 프로그램 메모리용 롬 콤포넌트 파일 변환 종료(RomConv.exe)
S2
롬 변환 프로그램을 실행시켜서 롬 소프트 IP 파일을 생성(inprom.vhd)
S3
MPU 코어 IP와 병합하여, CAD 툴로 회로 합성 및 검증
S4
룸 내장 MPU 소프트 IP (ET8051.vhd)
S5

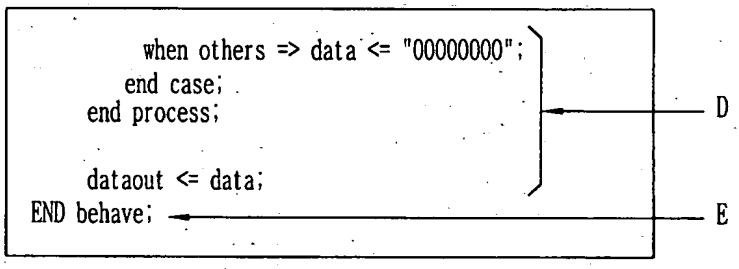
【도 2】


10
MPU 소프트 IP (ET8051.vhd)
30
MPU 코어 IP (ET8031.vhd)
20
롬 소프트 IP (Inprom.vhd)
31
중앙처리장치(CPU)
및
기타 블럭
Maddr
InPmDIn
16 비트
어드레스 버스
8 비트
명령어 버스
addr
dataout
0~64KB 크기의
MPU 구동용
프로그램 메모리
블럭

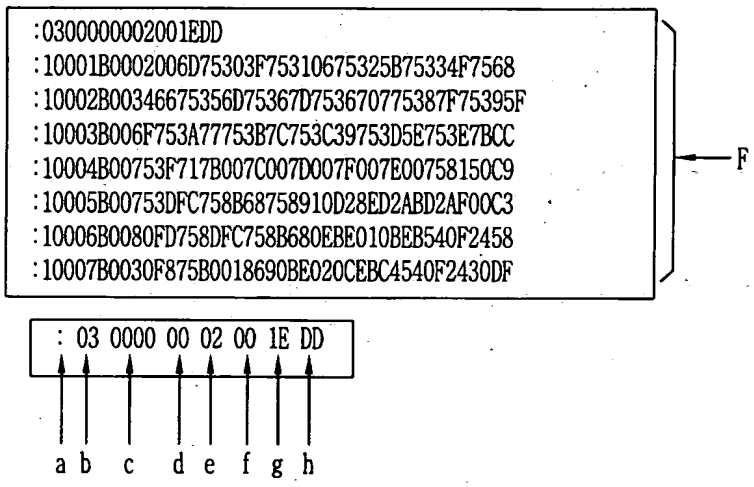
【도 3】


LIBRARY ieee;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
LIBRARY work;
USE work.et8051_pkg.all;
A
ENTITY inprom IS
PORT(
addr: IN std_logic_vector(15 downto 0) ;
dataout: OUT DATGBTYPE
);
END inprom;
B
ARCHICTECTURE behave OF inprom IS
signal data: DATGBTYPE;
begin
process(addr)
begin
case addr is
C

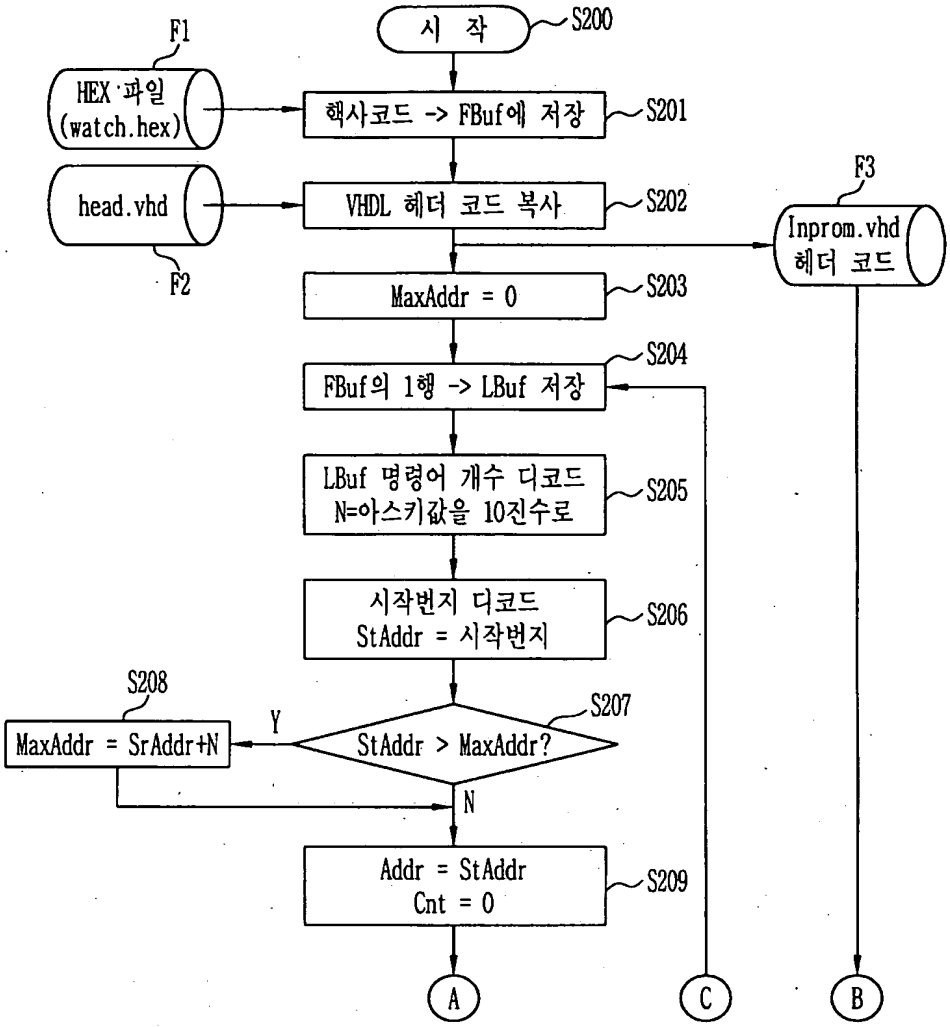
【도 4】


when others => data <= "00000000";
end case;
end process;
D
dataout <= data;
END behave;
E

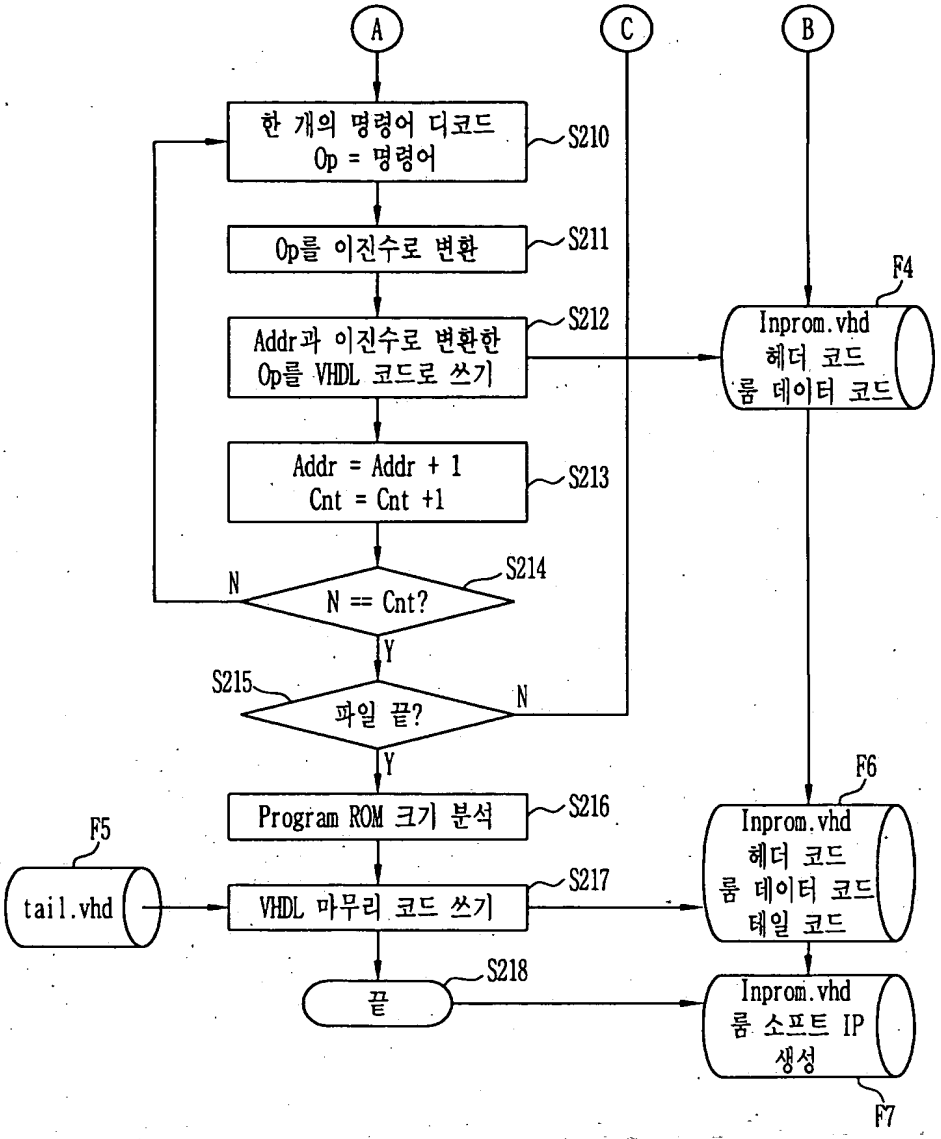
【도 5】


:0300000002001EDD
:10001B0002006D75303F75310675325B75334F7568
:10002B00346675356D75367D753670775387F75395F
:10003B006F753A77753B7C753C39753D5E753E7BCC
:10004B00753F717B007C007D007F007E00758150C9
:10005B00753DFC758B68758910D28ED2ABD2AF00C3
:10006B0080FD758DFC758B680EBE010BEB540F2458
:10007B0030F875B0018690BE020CEBC4540F2430DF
F
: 03 0000 00 02 00 1E DD
a b c d e f g h

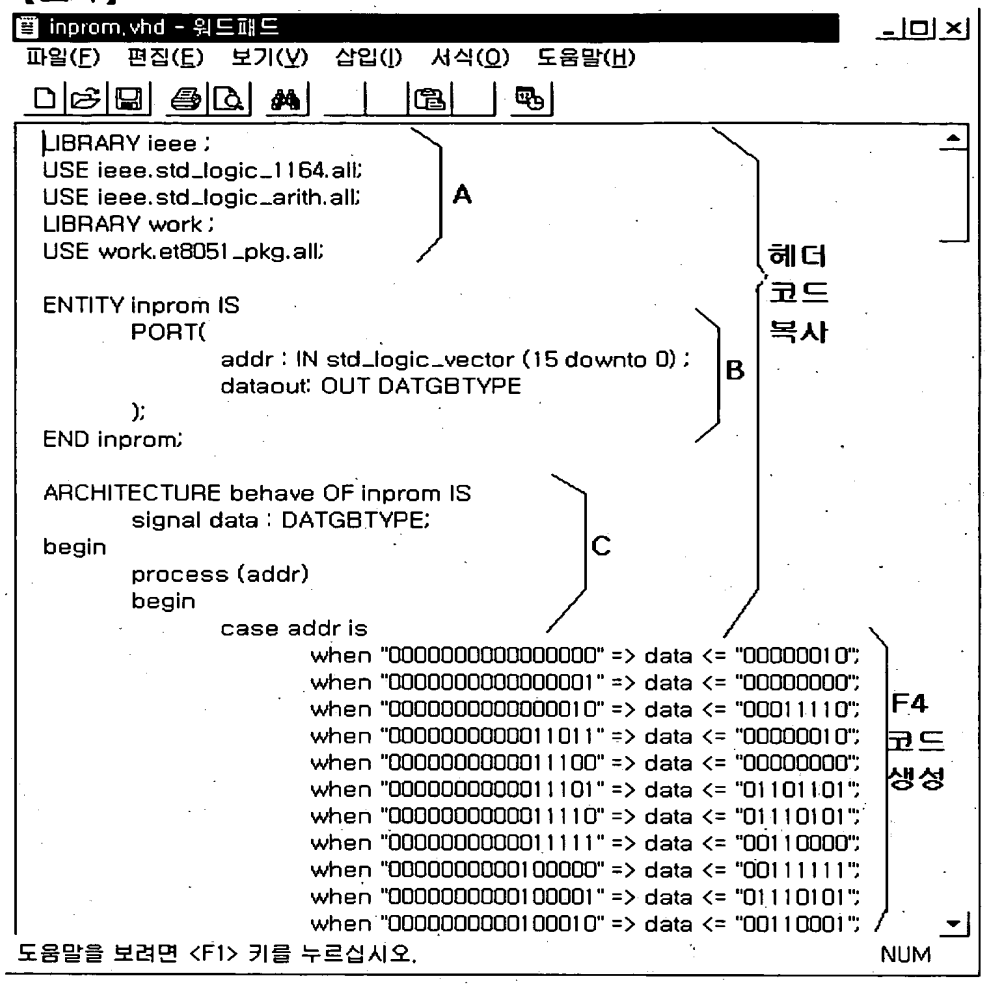
【도 6a】


시 작
S200
F1
HEX 파일
(watch.hex)
핵사코드 -> FBuf에 저장
S201
head.vhd
VHDL 헤더 코드 복사
S202
F3
Inprom.vhd
헤더 코드
F2
MaxAddr = 0
S203
S204
FBuf의 1행 -> LBuf 저장
LBuf 명령어 개수 디코드
N=아스키값을 10진수로
S205
시작번지 디코드
StAddr = 시작번지
S206
S208
S207
MaxAddr = SrAddr+N
Y
StAddr > MaxAddr?
N
Addr = StAddr
Cnt = 0
S209
A
C
B

【도 6b】


A
C
B
한 개의 명령어 디코드
Op = 명령어
S210
Op를 이진수로 변환
S211
Addr과 이진수로 변환한
Op를 VHDL 코드로 쓰기
S212
Addr = Addr + 1
Cnt = Cnt +1
S213
N
N == Cnt?
S214
Y
S215
파일 끝?
N
Y
Program ROM 크기 분석
S216
F5
tail.vhd
VHDL 마무리 코드 쓰기
S217
끝
S218
F4
Inprom.vhd
헤더 코드
롬 데이터 코드
F6
Inprom.vhd
헤더 코드
롬 데이터 코드
테일 코드
Inprom.vhd
롬 소프트 IP
생성
F7

【도 7】


inprom.vhd - 워드패드
파일(F) 편집(E) 보기(V) 삽입(I) 서식(O) 도움말(H)
LIBRARY ieee ;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
LIBRARY work ;
USE work.et8051_pkg.all;
A
헤더
코드
복사
ENTITY inprom IS
PORT(
addr : IN std_logic_vector (15 downto 0) ;
dataout: OUT DATGBTYPE
);
END inprom;
B
ARCHITECTURE behave OF inprom IS
signal data : DATGBTYPE;
begin
process (addr)
begin
C
case addr is
when "0000000000000000" => data <= "00000010";
when "0000000000000001" => data <= "00000000";
when "0000000000000010" => data <= "00011110";
when "0000000000011011" => data <= "00000010";
when "0000000000011100" => data <= "00000000";
when "0000000000011101" => data <= "01101101";
when "0000000000011110" => data <= "01110101";
when "0000000000011111" => data <= "00110000";
when "0000000000100000" => data <= "00111111";
when "0000000000100001" => data <= "01110101";
when "0000000000100010" => data <= "00110001";
F4
코드
생성
도움말을 보려면 <F1> 키를 누르십시오.
NUM